API

AdminLogin Controller

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Mvc;

using webApi.Data;

using webApi.Models;

namespace webApi.Controllers

{

[Route("api/[controller]")]

public class AdminController : Controller

{

private readonly MyDbContext \_context;

public AdminController(MyDbContext context)

{

\_context = context;

}

// GET api/values

[HttpGet]

public List<AdminLogin> Get()

{

var admn = \_context.AdminLogins.ToList();

return (admn);

}

// GET api/values/5

[HttpGet("{id}")]

public IActionResult Get(int id)

{

var admn = \_context.AdminLogins.Find(id);

if (admn == null)

{

return NotFound();

}

return Ok(admn);

}

// POST api/values

[HttpPost]

public IActionResult AdminLogin([FromBody]AdminLogin obj)

{

var list1 = \_context.AdminLogins.ToList();

var list = \_context.AdminLogins.ToList().Where(s => s.UserName == obj.UserName & s.Password == obj.Password);

if (list.Count() > 0)

return Ok(list);

return NotFound("Invalid Login details");

}

// PUT api/values/5

[HttpPut("{id}")]

public void Put(int id, [FromBody]string value)

{

}

// DELETE api/values/5

[HttpDelete("{id}")]

public void Delete(int id)

{

}

}

}

AdminLogin Model

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations.Schema;

using System.Linq;

using System.Threading.Tasks;

namespace webApi.Models

{

[Table("adminLogin")]

public class AdminLogin

{

[Column("adminId")]

public int ID { get; set; }

[Column("adminUserName")]

public String UserName { get; set; }

[Column("adminPassword")]

public String Password { get; set; }

}

}

ItemMaster Controller

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using webApi.Data;

using webApi.Models;

namespace webApi.Controllers

{

[Produces("application/json")]

[Route("api/itemMaster")]

public class itemMasterController : Controller

{

private readonly MyDbContext \_context;

public itemMasterController(MyDbContext context)

{

\_context = context;

}

// GET: api/itemMaster

[HttpGet]

public List<ItemMaster> Get()

{

var item = \_context.ItemMasters.Include(s=>s.unitMaster).Include(s=>s.typeMaster).Where(s => s.isActive == true).ToList();

return (item);

}

// GET: api/itemMaster/5

//[HttpGet("{id}", Name = "Get")]

[HttpGet("{id}")]

public string Get(int id)

{

return "value";

}

// POST: api/itemMaster

[HttpPost]

public IActionResult itemMaster([FromBody]ItemMaster obj)

{

obj.isActive = true;

if (obj.itemId != 0)

{

\_context.ItemMasters.Attach(obj);

\_context.ItemMasters.Update(obj);

}

else

\_context.ItemMasters.Add(obj);

\_context.SaveChanges();

return new ObjectResult("Unit Added Successfully");

}

// PUT: api/itemMaster/5

[HttpPut("{id}")]

public void Put(int id, [FromBody]string value)

{

}

// DELETE: api/ApiWithActions/5

[HttpDelete("{id}")]

public IActionResult Delete(int id)

{

var item = \_context. ItemMasters.SingleOrDefault(m => m.itemId == id);

if (item == null)

{

return NotFound();

}

item.isActive = false;

\_context. ItemMasters.Update(item);

\_context.SaveChanges();

return Ok(item);

}

}

}

Item master Model

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

using System.Linq;

using System.Threading.Tasks;

namespace webApi.Models

{

[Table("itemMaster")]

public class ItemMaster

{

[Key]

[Column("itemId")]

public int itemId { get; set; }

public string itemName { get; set; }

public int itemQuantity { get; set; }

public double itemPrice { get; set; }

[ForeignKey("unitId")]

public int unitId { get; set; }

public unitMaster unitMaster { get; set; }

[ForeignKey("typeId")]

public int typeId { get; set; }

public typeMaster typeMaster { get; set; }

public bool isActive { get; set; }

}

}

DbContextFile

using Microsoft.EntityFrameworkCore;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using webApi.Models;

namespace webApi.Data

{

public class MyDbContext : DbContext

{

public MyDbContext(DbContextOptions<MyDbContext> options) : base(options)

{ }

public DbSet<AdminLogin> AdminLogins { get; set; }

public DbSet<ItemMaster> ItemMasters { get; set; }

public DbSet<unitMaster> unitMasters { get; set; }

public DbSet<typeMaster> typeMasters { get; set; }

public DbSet<areaMaster> areaMasters { get; set; }

public DbSet<societyMaster> societyMasters { get; set; }

public DbSet<cityMaster> cityMaster { get; set; }

public DbSet<supplierMaster> supplierMasters { get; set; }

public DbSet<flatMaster> flatMasters { get; set; }

public DbSet<customerMaster> customerMasters { get; set; }

}

}

Appsetting Connection string

{

"ConnectionStrings": {

"DefaultConnection": "Data Source=.\\SQLEXPRESS;Initial Catalog=db\_boom;Integrated Security=True"

},

"Logging": {

"IncludeScopes": false,

"Debug": {

"LogLevel": {

"Default": "Warning"

}

},

"Console": {

"LogLevel": {

"Default": "Warning"

}

}

}

}

Startup File

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Builder;

using Microsoft.AspNetCore.Hosting;

using Microsoft.EntityFrameworkCore;

using Microsoft.Extensions.Configuration;

using Microsoft.Extensions.DependencyInjection;

using Microsoft.Extensions.Logging;

using Microsoft.Extensions.Options;

using webApi.Data;

using webApi.Models;

namespace webApi

{

public class Startup

{

public Startup(IConfiguration configuration)

{

Configuration = configuration;

}

public IConfiguration Configuration { get; }

// This method gets called by the runtime. Use this method to add services to the container.

public void ConfigureServices(IServiceCollection services)

{

services.AddMvc();

services.AddCors();

services.AddDbContext<MyDbContext>(options => options.UseSqlServer(Configuration.GetConnectionString("DefaultConnection")));

}

// This method gets called by the runtime. Use this method to configure the HTTP request pipeline.

public void Configure(IApplicationBuilder app, IHostingEnvironment env)

{

if (env.IsDevelopment())

{

app.UseDeveloperExceptionPage();

}

app.UseCors(builder => builder

.AllowAnyOrigin()

.AllowAnyMethod()

.AllowAnyHeader()

.AllowCredentials());

app.UseMvc();

}

}

}

unitMaster

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using webApi.Data;

using webApi.Models;

namespace webApi.Controllers

{

[Produces("application/json")]

[Route("api/unitMaster")]

public class unitMasterController : Controller

{

private readonly MyDbContext \_context;

public unitMasterController(MyDbContext context)

{

\_context = context;

}

// GET: api/unitMaster

[HttpGet]

public List<unitMaster> Get()

{

var admn = \_context.unitMasters.Where(s => s.isActive == true).ToList();

return (admn);

}

// GET: api/unitMaster/5

[HttpGet("{id}", Name = "Get")]

public string Get(int id)

{

return "value";

}

// POST: api/unitMaster

[HttpPost]

public IActionResult unitMaster([FromBody]unitMaster obj)

{

obj.isActive = true;

if (obj.unitId != 0)

{

\_context.unitMasters.Attach(obj);

\_context.unitMasters.Update(obj);

}

else

\_context.unitMasters.Add(obj);

\_context.SaveChanges();

return new ObjectResult("Unit Added Successfully");

}

// PUT: api/unitMaster/5

[HttpPut("{id}")]

public void Put(int id, [FromBody]string value)

{

}

// DELETE: api/ApiWithActions/5

[HttpDelete("{id}")]

public IActionResult Delete(int id)

{

var item = \_context.unitMasters.SingleOrDefault(m => m.unitId == id);

if(item == null)

{

return NotFound();

}

item.isActive = false;

\_context.unitMasters.Update(item);

\_context.SaveChanges();

return Ok(item);

}

}

}

unitMaster Model

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

using System.Linq;

using System.Threading.Tasks;

namespace webApi.Models

{

[Table("unitMaster")]

public class unitMaster

{[Key]

[Column("unitId")]

public int unitId { get; set; }

public string unitName { get; set; }

public string unitDescription { get; set; }

public bool isActive { get; set; }

//public ICollection<ItemMaster> itemMasters { get; set; }

}

}

Customer Master controller

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using webApi.Data;

using webApi.Models;

namespace webApi.Controllers.customerMasters

{

[Produces("application/json")]

[Route("api/customerMasters")]

public class customerMastersController : Controller

{

private readonly MyDbContext \_context;

public customerMastersController(MyDbContext context)

{

\_context = context;

}

// GET: api/customerMasters

[HttpGet]

public List<customerMaster> Get()

{

var customer = \_context.customerMasters.Include(s => s.cityMaster).Include(s => s.societyMaster).Include(s => s.areaMaster).Include(s => s.flat).ToList();

return (customer);

}

// GET: api/customerMasters/5

[HttpGet("{id}")]

public async Task<IActionResult> GetcustomerMaster([FromRoute] int id)

{

if (!ModelState.IsValid)

{

return BadRequest(ModelState);

}

var customerMaster = await \_context.customerMasters.SingleOrDefaultAsync(m => m.custId == id);

if (customerMaster == null)

{

return NotFound();

}

return Ok(customerMaster);

}

// PUT: api/customerMasters/5

[HttpPut("{id}")]

public async Task<IActionResult> PutcustomerMaster([FromRoute] int id, [FromBody] customerMaster customerMaster)

{

if (!ModelState.IsValid)

{

return BadRequest(ModelState);

}

if (id != customerMaster.custId)

{

return BadRequest();

}

\_context.Entry(customerMaster).State = EntityState.Modified;

try

{

await \_context.SaveChangesAsync();

}

catch (DbUpdateConcurrencyException)

{

if (!customerMasterExists(id))

{

return NotFound();

}

else

{

throw;

}

}

return NoContent();

}

// POST: api/customerMasters

[HttpPost]

public async Task<IActionResult> PostcustomerMaster([FromBody] customerMaster cust)

{

if (!ModelState.IsValid)

{

return BadRequest(ModelState);

}

\_context.customerMasters.Add(cust);

await \_context.SaveChangesAsync();

return CreatedAtAction("GetFlat", new { id = cust.flatId }, cust);

}

// DELETE: api/customerMasters/5

[HttpDelete("{id}")]

public async Task<IActionResult> Deletecustomer([FromRoute] int id)

{

if (!ModelState.IsValid)

{

return BadRequest(ModelState);

}

var cust = await \_context.customerMasters.SingleOrDefaultAsync(m => m.custId == id);

if (cust == null)

{

return NotFound();

}

\_context.customerMasters.Remove(cust);

await \_context.SaveChangesAsync();

return Ok(cust);

}

private bool customerMasterExists(int id)

{

return \_context.customerMasters.Any(e => e.custId == id);

}

}

}

**Customer Master Model**

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

using System.Linq;

using System.Threading.Tasks;

namespace webApi.Models

{

[Table("customerMaster")]

public class customerMaster

{

[Key]

[Column("custId")]

public int custId { get; set; }

public string custName { get; set; }

[ForeignKey("cityId")]

public int cityId { get; set; }

public cityMaster cityMaster { get; set; }

[ForeignKey("areaId")]

public int areaId { get; set; }

public areaMaster areaMaster { get; set; }

[ForeignKey("societyId")]

public int societyId { get; set; }

public societyMaster societyMaster { get; set; }

[ForeignKey("flatId")]

public int flatId { get; set; }

public flatMaster flat { get; set; }

public string custMobNo1 { get; set; }

public string custMobNo2 { get; set; }

public string custGeoLocation { get; set; }

public bool isActive { get; set; }

}

}